A METHOD FOR CONTROLLING SKEW ON LINKED SURFACES[1]

Robert A. Kerr[2], Steven E. Benzley[3], David R. White[4], Scott Mitchell[5]

[2]*Brigham Young University, Provo, UT, U.S.A. kerrr@et.byu.edu*
[3]*Brigham Young University, Provo, UT, U.S.A. seb@byu.edu*

Abstract

A new method for lessening skew in mapped and submapped meshes is presented. This new method involves progressive subdivision of a surface into loops consisting of four sides. Using these loops, matching constraints can be set on the curves of the surface, which will propagate interval assignments across the surface, allowing a mesh with a better skew metric to be generated.

Keywords: **mesh generation, skew measure, virtual vertex, mapped mesh, mesh quality**

1. INTRODUCTION

One of the desired outcomes of automatic meshing is good element quality. It is well understood that a good quality mesh yields better results than does one with bad quality [1,2]. There are many different measures that allow the quantification of "good quality". Among this group [3] are such metrics as jacobian, aspect ratio, taper, warpage, element area, stretch, maximum angle, minimum angle, oddy condition number, and skew. Skew, although not a large factor in many types of meshing problems, can propagate across models which consist of many linked parts. This propagation can cause severe deterioration of the mesh quality on the model. This paper will concentrate on the problem of controlling skew across linked surfaces.

2. THE SKEW PROBLEM

Skew is defined as the maximum absolute value of the cosine of the angle between edges at the center of the quadrilateral. In other words, it is the cosine of the angles formed by the two lines which pass through the midpoints of the sides of the quadrilateral [4]. In Figure 1, a quadrilateral element is shown, with one of the angles labeled as angle A. The absolute value of the cosine of this angle would be one of the four possible measures of skew for this element. The

[1] This work was partly funded by Sandia National Laboratories, operated for the U.S. DOE under contract No. DE-AL04-94AL8500. Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the U.S. DOE.

[4] drwhite@sandia.gov David R. White works at Sandia National Laboratories, operated for the U.S. DOE under contract No. DE-AL04-94AL8500. Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the U.S. DOE.

[5] samitch@sandia.gov Scott Mitchell was supported by the Mathematical, Information and Computational Sciences Division of the U.S. Department of Energy, Office of Energy Research.

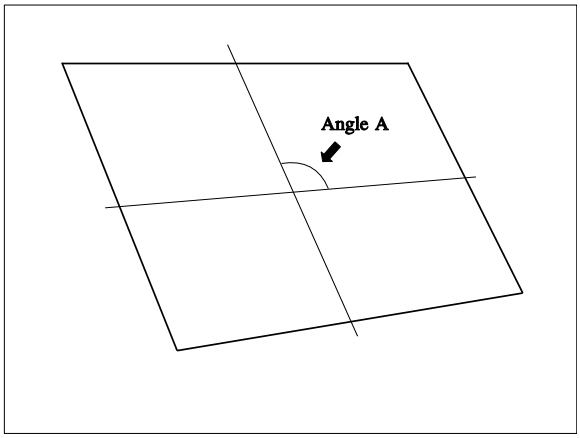

Figure 1: Quadrilateral mesh element showing representative angle 'A'.

cosines of the other three angles would be calculated, and the maximum of those four cosines would be the measure of skew for this face. As can be seen, the value of skew ranges between 0.0 (because of the absolute value operation) and 1.0, with the optimal value being 0. Mapped meshes, by their nature, depend

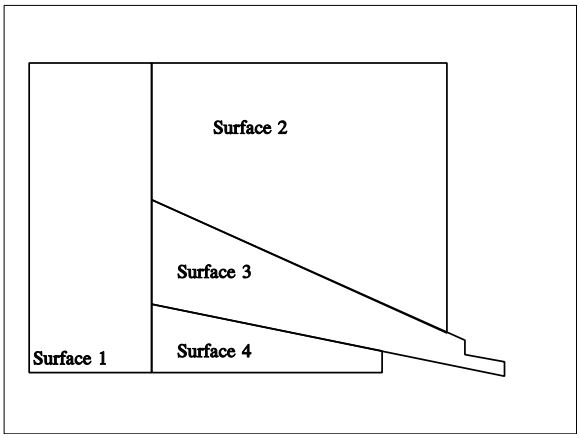

Figure 2: Multiply-connected set of surfaces which could lead to skew problems.

on propagation of interval assignments [5,6]. Skew usually isn't a problem in small, simple models, however when many volumes are multiply-connected, as shown in Figure 2, interval assignments propagate throughout the model. This propagation of interval assignments can lead to skew problems. In Figure 2, interval settings on the ends of surfaces 2, 3, and 4 will propagate across those surfaces and affect the right-hand side of surface 1. Surface 1 will then have to have its left-hand edge set to the same number of intervals as is on the right-hand side. If one of the right-hand side curves has a comparatively high interval count, the mesh on surface 1 could become greatly skewed. Figure 3 shows, for example, a

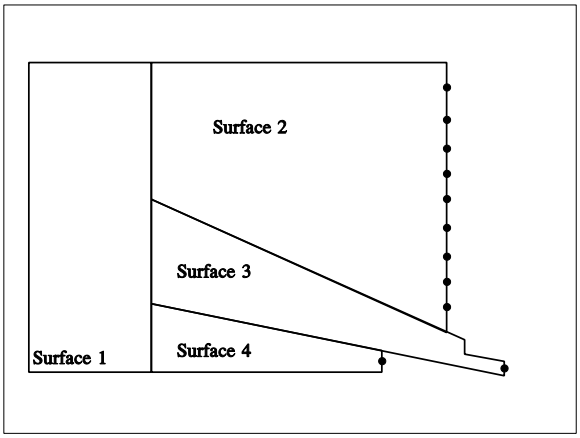

Figure 3: Set of surfaces with interval assignments on right-hand sides.

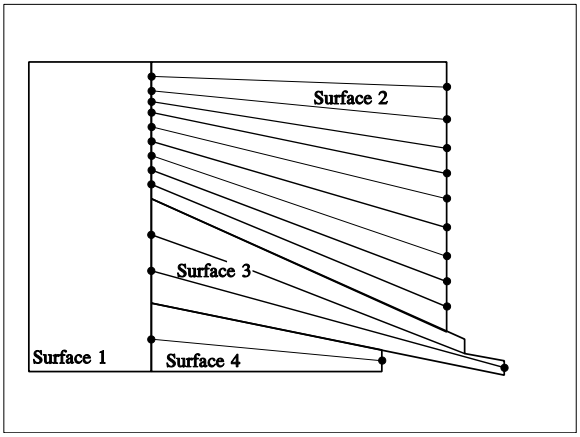

Figure 4: Interval propagation across surfaces.

possible set of intervals for the right-hand side of surfaces 2, 3, and 4. These intervals will propagate across the surfaces, as shown in Figure 4. After propagating across the surfaces, the intervals on the common edge between surface 1 and the other three surfaces are firmly set. A mesh then needs to be generated on surface 1. Most mesh generating software assigns equally-spaced intervals as the default. Therefore, the left-hand side interval assignment on surface 1 would look like that shown in Figure 5. The final meshing of surface 1 would then be done, yielding a final mesh as is shown in Figure 6. As can be seen, the mesh on this surface would result in a large degree of skew. What is required to reduce skew is to develop a method that will transfer the interval assignments from one side of the surface to the other in a manner that will preserve the relative interval spacings on different sections of the curves. Intervals on opposite sides cannot just be set to have the same intervals, because these opposite edges may be of different lengths. Manually setting intervals on

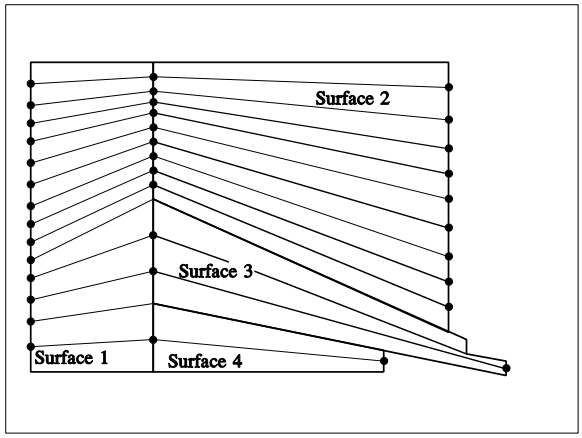

Figure 5: Default interval assignment on surface 1.

surface 1 would still yield that same mesh shown in Figure 6. As can be seen, this is a severely limited solution to the problem of controlling skew-it isn't better than what would be done automatically. It is this circumstance which has prompted the development of the skew control algorithm.

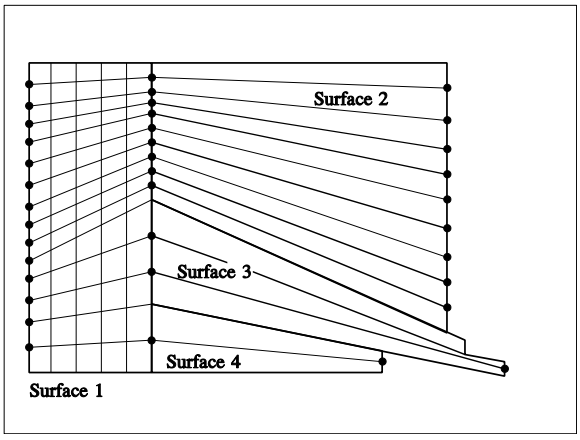

Figure 6: Final skewed mesh on surface 1.

3. THE SKEW CONTROL ALGORITHM

The purpose of this algorithm is to manage the interval settings on surfaces that will probably exhibit unacceptable skew. The Skew Control algorithm is intended for use previous to meshing, although the user is allowed to assign approximate sizes and intervals. If the user has specified this information, the algorithm will respect those settings. Given a surface such as that discussed above, the skew control algorithm developed here will partition the edges of the surface and set up a system of matching edges across a surface or multiple surfaces. Once this is done, the interval count on corresponding edges is set to be equal, so that a created mesh has little or no skew. Instead of the skewed mesh in Figure 6, the final mesh would result as depicted in Figure 7.

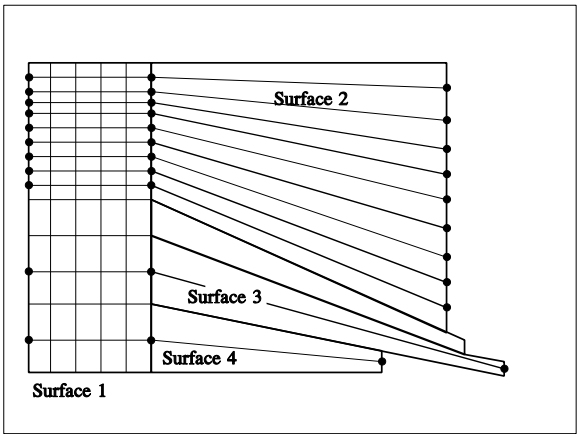

Figure 7: Surfaces showing a skew-controlled mesh.

The skew control algorithm has the following 7 steps:

1. Approximate the affected surfaces with pseudo geometry.

2. Create a loop of edges around the base surface.

3. Find the smallest projection on the surface.

4. Separate this small feature from the rest of the surface using a pseudo edge.

5. Continue separating small sections until all loops consist of only four edges.

6. Step through loops setting up interval assignments for opposite edges.

7. Clean up pseudo geometry.

These steps will be explained one-by-one with representative illustrations.

1. Approximate the affected surfaces.

The skew control algorithm depends on an approximation of the surfaces to be meshed. These surfaces will be referred to as the base surfaces. The same set of surfaces shown in Figure 2 will be used in this example. The algorithm uses pseudo geometry known as skew control entities for this approximation.

The curves and vertices which make up the real surface are used as templates to create skew control edges and skew control vertices. These skew control entities are the basis for almost all the work until the algorithm reaches step 7. Skew control entities only need to hold a little information. Each skew control edge knows what vertices define it, and each skew control vertex knows its position in three-space and its type, which will be defined later. Because of this sparsity of information, the memory overhead in using these pseudo-entities is small.

2. Create a loop of edges around the base surfaces.

The skew control edges and vertices that have been created from the base surfaces are now placed in lists which maintain their order. Each base surface is approximated by one list whose edges and vertices form a loop. These loops, which are the base for this algorithm, define the base surfaces throughout the rest of the algorithm. This example set of base surfaces would translate into four loops.

3. Find the smallest projection in the surface.

The skew control algorithm implements a type of "blocking" subdivision. "Blocking" refers to the process of dividing up the surface into blocks, or four-sided figures. This blocking algorithm starts with the smallest areas, filling them with blocks, then expanding to the larger ones. The algorithm could start with larger areas, and work to smaller ones, but by starting with the smallest, the problem of intersection checking becomes much less pronounced. The starting step is to find the smallest end. For some geometries, this is not an easy location to find, and much depends on the definition of "end". An "end" is defined as a set of edges that are bounded by two vertices which are known as End_Types. As can be seen, the type of the skew control vertices has a great impact on the definition of an end. There are four types of vertices, based on the angle of the edges which share the vertex:

1. End_Type, with an angle close to 90°.

2. Side_Type, with an angle close to 180°.

3. Corner_Type, with an angle close to 270°.

4. Reversal_Type, with an angle close to 360° [5].

Figure 8 shows all four types. A skew control vertex is assigned a type based on the type of the underlying real vertex. If there is no underlying vertex, the type is computed based on the angle of the connected skew

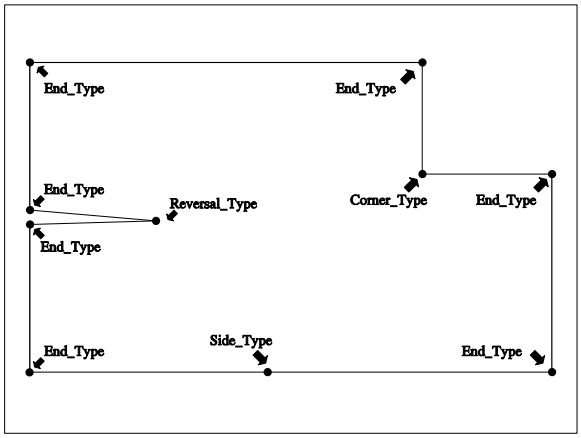

Figure 8: Surface showing the four vertex types.

control edges. The base surface's loop is searched for the "end" that is shortest, and that end is used for the next operation. As shown in Figure 9 the far right-hand end will be picked

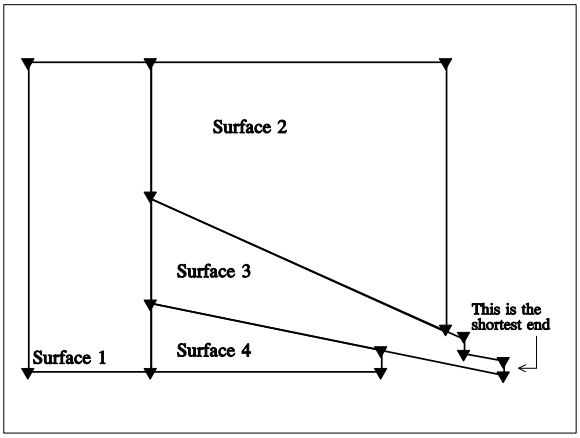

Figure 9: Surfaces showing which end will be picked first. Vertices are indicated by the dark triangles.

4. Separate the smallest projection from the rest of the loop.

When the shortest end has been found, the algorithm then looks at the two edges that form the sides of the block. At this point there are two possibilities: either one side will be longer than the other, or the two will be the same length (within a tolerance). If the first case applies, the algorithm splits the longer of the two sides so that the two sides are equal. Of course, this splitting can happen within a range of lengths, depending on local geometry and the desired behavior. If the pseudo edge to be split has an underlying geometry edge, that geometry edge is split too, and a virtual vertex is inserted. After the split of the pseudo

edge, this first case becomes identical to the second case. When two vertices are at an equal distance from the end, a new pseudo edge is created between the two vertices. The edges and vertices comprising the block being replaced are then removed from the old loop, and the new edge is inserted in their place. A new loop is created using the newly independent edges and vertices and the new edge, as shown in Figure 10. Where once there was one loop defining the surface, now there are two. An important feature of this new loop is that it consists of only four edges.

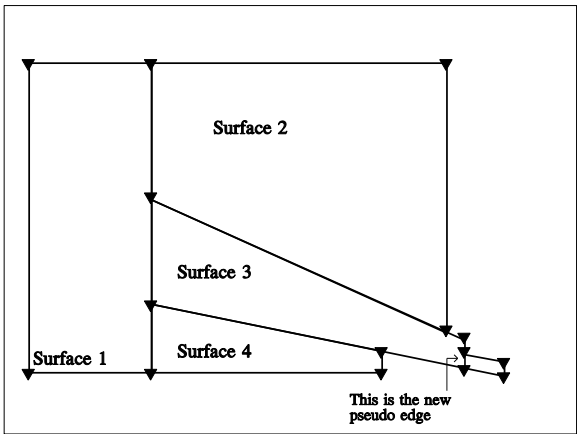

Figure 10: Surfaces after splitting curve into two curves and inserting a new pseudo edge. Note the new virtual vertex.

5. Continue separating small sections until all loops consist of only four edges.

Now there are five loops to consider, instead of the previous four. Each of these loops is checked to see if they have more than four edges. If one is found, the previous step is repeated and the next, smallest projection is separated from that loop and put into its own loop. Now, because of the possibility of an edge being in two loops at the same time, it is vital that the loops stay current regarding which edges belong to them. It would cause a severe problem if an edge were split but the loops containing the old edge still had pointers to that obsolete edge. Pointers to these owning loops are of necessity another piece of data contained in each skew control edge. Each edge keeps a list of loops it belongs to. With this information, when an edge is subdivided, as in step 4, the algorithm can access this list of loops and update each one of them so that they contain the correct information. A side effect of this is that a loop that previously only had four edges might end up having more through the splitting of one of its included edges. This necessitates stepping through the set of loops multiple times until all the loops have only four edges. An example of this increase in edges is seen in Figure 11 where the right-side curve of the center left-hand loop (which is shown in bold in the figure) is being subdivided. The

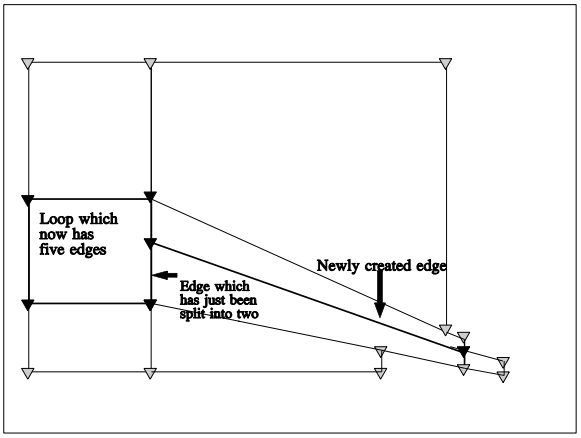

Figure 11: Surfaces showing loop about to be subdivided.

next time that loop is examined, it will be processed again, because of the increase in number of edges. This is shown in Figure 12 where all the needed subdivisions have taken place.

6. Step through loops setting up interval assignments for opposite edges.

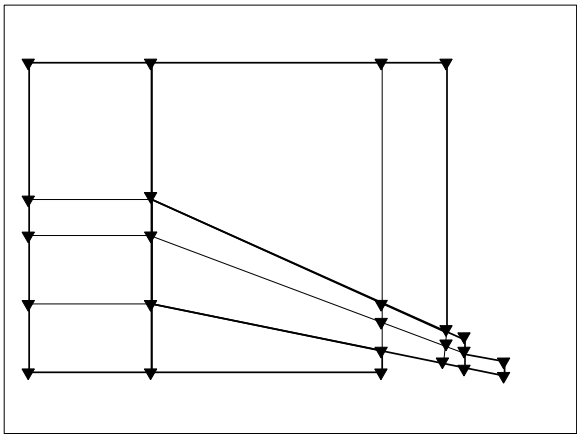

Figure 12: Surfaces showing final subdivision of loops.

Once all of the loops satisfy the requirement of having only four edges, the setting of constraints can be done. Each loop is examined to find an edge that has an underlying geometry edge (or owner). This edge is then marked and the loop is searched for the edge

opposite the marked one. If the opposite edge has an underlying owner, then those two edges are set to have equal intervals. If this opposite edge does not have an owner, the algorithm asks the edge for its list of owning loops and these loops are searched in the same manner. In this way, geometry edges that are at either end of a series of loops are found and then set to have equal intervals. Figure 13 shows the final state of the example surfaces, with each set of labeled edges having the same interval setting. Because each edge

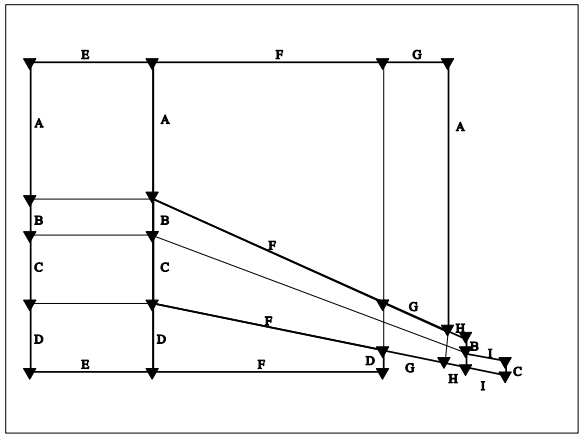

Figure 13: Surfaces showing final subdivisions with interval groups labeled.

can belong to any number of loops, the interval assignment on one edge may propagate to quite a few different edges on many different surfaces, but this is handled transparently. The algorithm doesn't know or care if the edges it is dealing with are on the same surface or different ones, all it sees are loops and owners.

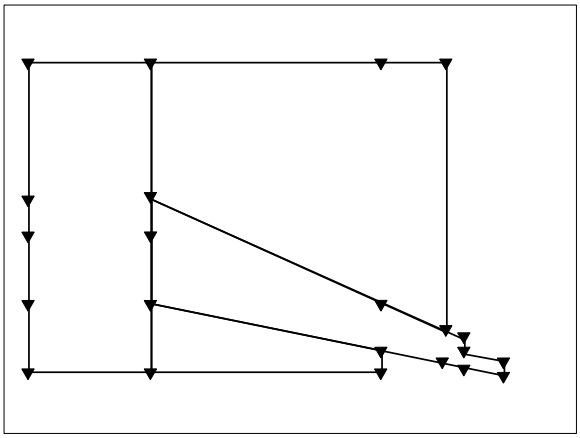

Figure 14: Surfaces after skew control cleanup.

7. Clean up pseudo geometry.

After the interval assignment has been done, the skew control entities can be safely deleted. The only changes done to the underlying geometry are that where edges had to be split, there are now two edges and a vertex (Figure 14), and interval assignments have been made to all the curves in the base surfaces.

4. THE SKEW CONTROL ALGORITHM: RESULTS

The model shown in Figure 15 has quite a few multiply-connected surfaces that can cause skew. For example, the lower right-hand surface from this model,

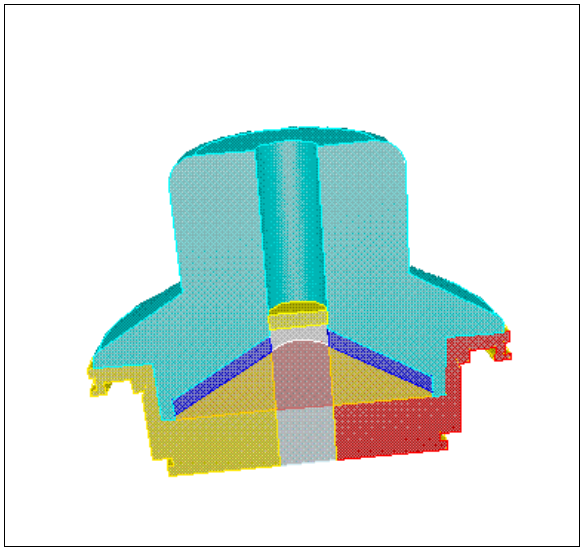

Figure 15: Model with potential skew problems.

shown in Figure 16, is rather complicated, with many other surfaces touching it. If meshed without skew control, using a submapping algorithm, the mesh would be created as shown in Figure 17. As can be seen, there is quite a bit of skew evident on the surface. However, if the skew control algorithm is applied to this surface, the resulting mesh demonstrates much less skew. The skew control algorithm processes this surface, inserting virtual vertices in places it deems appropriate. Since the user has already requested a certain mesh size on the model, this is taken into account in the final mesh, through a mechanism of transferring interval assignments onto newly created curves. The surface in question now has new vertices, as shown in Figure 18.

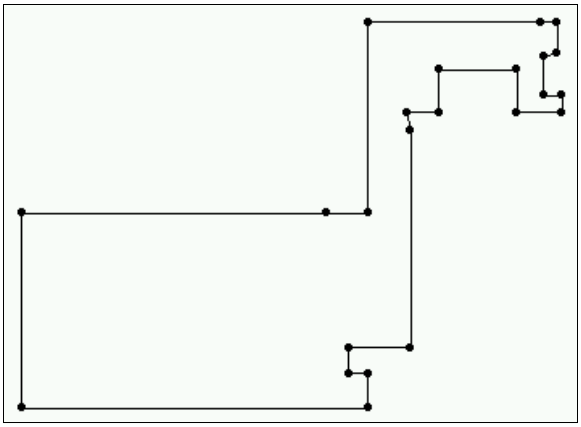

Figure 16: Hooked surface before meshing.

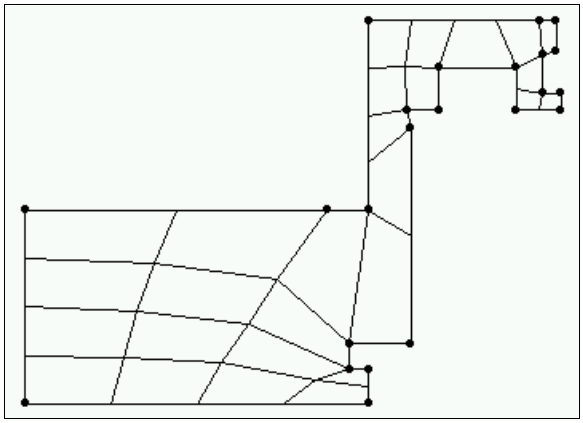

Figure 17: Hooked surface after meshing without skew control.

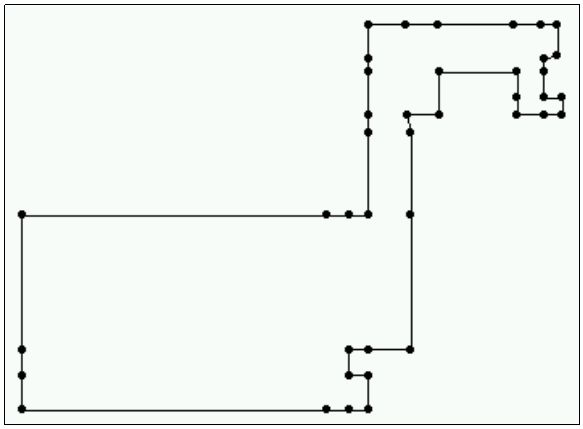

Figure 18: Hooked surface after processing by the skew control algorithm.

The mesh that results from this changed surface is shown in Figure 19. As this shows, the mesh is noticeably less skewed. Use of the skew control algorithm does result in an increase in the number of edges and vertices in the model, but this has not

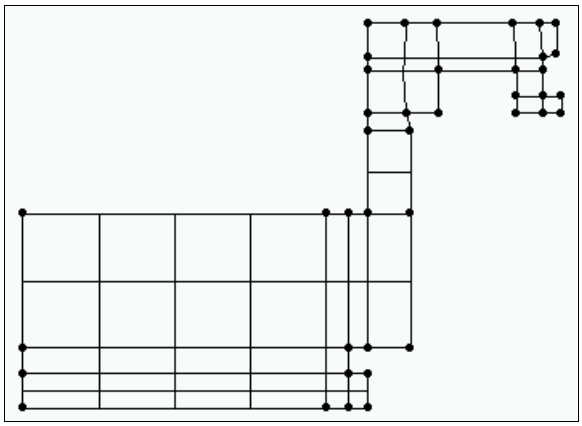

Figure 19: Hooked surface after meshing with skew control.

proven to be a major concern. The mesh of the Hooked surface is of good quality, and the measure of skew has decreased dramatically, as is shown in Table 1. Another example of skew control is shown in

Table 1: Comparison of skew for hooked surface with and without skew control.

Hooked Surface	Non-controlled	Controlled
Maximum Skew	0.7081	0.1139
Minimum Skew	0.02564	0.00
Average Skew	0.2384	0.01282

Figures 20 and 21. As can be seen, this is a model similar to the example which was used to demonstrate the steps of the algorithm. There are some minor differences in the model, and hence in the final mesh-this is due to the difficulty of drawing the exact figure using presentation software. Figure 20 demonstrates the mesh that is generated without having applied the skew control algorithm. This surface was meshed by a submapping algorithm, after the intervals on most of the sides were set. Although this paper concentrated on the surface on the far left, called surface 1, it is also obvious that there are skew

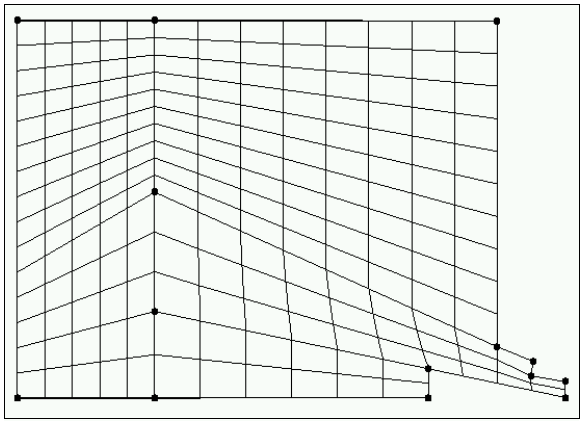

Figure 20: Skewed mesh generated on linked surfaces.

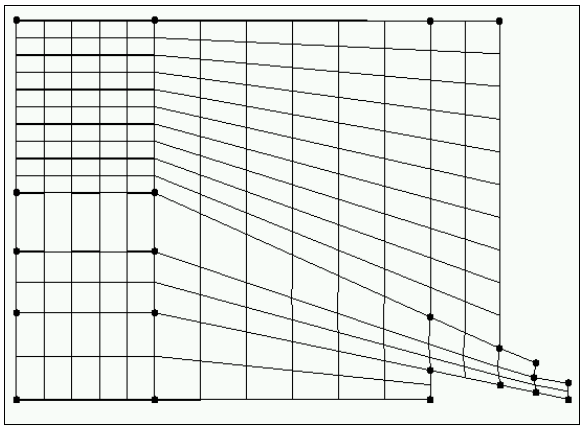

Figure 21: Less-skewed mesh generated with skew control algorithm on linked surfaces.

problems in other surfaces too. Figure 21 demonstrates the model with a skew controlled mesh. As is shown, there is very little skew on surface 1, although the other surfaces, because of their inherently skewed nature, still have skewed meshes. Table 2 compares the values of skew obtained with and without skew control. They are comparable to the values shown for the previous example.

5. THE SKEW CONTROL ALGORITHM AND NON-PLANAR SURFACES

Non-planar surfaces present a particular problem for the skew control algorithm. Because the underlying geometry is not flat, the approximations done in creating skew control entities can lead to errors in the creation of new vertices. Because of this, the decision was made to treat non-planar surfaces in a different

Table 2: Comparison of skew for linked surfaces with and without skew control.

Linked Surfaces	Non-controlled	Controlled
Maximum Skew	0.4847	2.010×10^{-14}
Minimum Skew	0.02915	5.551×10^{-17}
Average Skew	0.2750	5.793×10^{-15}

manner. Instead of creating the four-sided loops that are used to subdivide regular planar surfaces, the algorithm propagates vertices to all possible curves. Each curve on a surface is classified as either a Positive I, Negative I, Positive J or Negative J curve. Then the vertices that are on the I curves are propagated to all the other I curves that encompass the position of the vertex in question. For example, if a vertex is located at position 5 in I-space, all curves that cross the 5 position will have a virtual vertex created at that point. The vertices on the J curves are propagated in the same manner. This design will succeed in locating vertices where they are needed, but it can also lead to a greater number of virtual vertices being created than is strictly necessary. It is expected that most non-planar surfaces will not be so complicated that they will cause an excess of vertex creation.

6 THE SKEW CONTROL ALGORITHM: FURTHER WORK

Although the skew control algorithm has been seen to provide good results in many cases, there are some issues that need to be addressed to make this a more widely applicable tool. Many of the remaining areas of debate exist because of particular design decisions that need to be made.

- The skew control algorithm only works on surfaces that are submappable, ie. of blocky, roughly four-sided sections or subsections. This is a limitation that will probably stand, because the purpose of the algorithm is to enhance such algorithms as submapping--it is not meant to be used for surfaces that would need to be meshed with an unstructured mesh.

- Biased intervals settings are not propagated correctly onto split curves. While general interval settings are handled automatically, the biasing is not preserved.

- Surfaces with more than one loop, ie. surfaces with holes in them, are not handled correctly. These types of surfaces will need to be modified before the skew control algorithm is called. The addition of an automatic surface cracker, an algorithm that will create a curve connecting an inner loop to an outer loop, will convert two-loop surfaces into one-loop surfaces. This type of surface will then become a one-loop case with a doubled edge.

- One-looped surfaces with doubled edges, as mentioned above, are not handled correctly. These surfaces are defined by a loop of edges in which one edge is encountered more than once. The skew control algorithm does not correctly split this type of edge, and more work needs to be done to enable this ability.

Another area of focus that deserves to be examined is the creation of virtual vertices. By creating vertices, the algorithm is able to easily set up the constraint equations for the final model. As can be seen by comparing Figures 16 and 18, the skew control algorithm does tend to insert a multitude of vertices. Although this creation of vertices hasn't proven to affect the model excessively, it is a goal of this research to develop a different way of setting up the constraint equations. It is hoped that through use of the curve morphing algorithm [7], an algorithm that duplicates a curve mesh or set of curve meshes onto a target curve or set of curves, the skew control algorithm will be able to correctly transfer the mesh from one side of a surface to another without the use of virtual vertices.

Perhaps the final area is that of testing. This tool needs to be tested extensively to decide such questions as tolerancing of curve intersections, and where to place target vertices on the target curve. In cases such as curved sides, the vertices need to have a better apparatus for deciding their final location--this will depend on the general shape of the loops.

In any case, the skew control algorithm seems to be a good area for further research in the hopes of lessening skew propagation caused by interval assignments, which will make large, complicated models much easier to mesh satisfactorily.

References

[1] Babuska, I., and Aziz, A., "On the Angle Condition in the Finite Element Method", SIAM Journal on Numerical Analysis, 13:214-226, 1976.

[2] Fried, I., "Condition of Finite Element Matrices Generated from Nonuniform Meshes", AIAA Journal, 10:219-22, 1972.

[3] Canann, S.A., Tristano, J.R., and Staten, M.L., "An Approach to Combined Laplacian and Optimization-Based Smoothing for Triangular, Quadrilateral, and Quad-Dominant Meshes", Proceedings of the 7th International Meshing Roundtable, Sandia National Laboratories, Oct., 1998.

[4] Robinson, J., "CRE Method of Element Testing and the Jacobian Shape Parameters", Eng. Comput., Vol. 4, No. 2, p113-118, June 1987.

[5] White, D.R., "Automatic, Quadrilateral and Hexahedral Meshing of Pseudo-Cartesian Geometries using Virtual Decomposition," Master's Thesis, Brigham Young University, August 1996.

[6] Mitchell, S.A., "High Fidelity Interval Assignment", Proceedings, 6th International Meshing Roundtable, SNL, Albuquerque, N.M., Oct 1997, 33-44.

[7] Kerr, R.A., "Improvement of Surface Meshes by Use of the Skew Control and Curve Morphing Algorithms," Master's Thesis in Progress, Brigham Young University, December 1999.